VŠB – Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra telekomunikační techniky

Implementace GPSd-NG klienta a serveru na platformě Android

GPSd-NG Client/Server Implementation on Android Platform

2012 **Michal Kika**

VŠB - Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra telekomunikační techniky

Zadání bakalářské práce

Student: **Michal Kika**

Studijní program: B2647 Informační a komunikační technologie

Studijní obor: 2612R059 Mobilní technologie

Téma: Implementace GPSd-NG klienta a serveru na platformě Android
GPSd-NG Client/Server Implementation on Android Platform

Zásady pro vypracování:

Sdílení dat o poloze z GPS je i dnes tématem, které je stále aktuální a nabízí celou řadu možných způsobů využití, ať již se jedná o vzdálený záznam pohybu, nebo potřebu sdílet tato data se zařízením, neobsahujícím GPS modul (např. některé levnější tablety na bázi Android OS). Cílem této práce je tvorba mobilní aplikace, která bude poskytovat jak serverovou tak klientskou část, potřebnou ke sdílení informací o poloze. Pro tento účel bude využit protokol, navržený pro GPSd.

1. Nastudujte oba typy protokolů využívaných GPSd, strukturu NMEA sentencí, a práci s Bluetooth (android.bluetooth) a určování polohy (android.location) v rámci Android API.
2. Vyhledejte aplikace, implementující serverovou či klientskou komponentu, používající tento protokol.
3. Analyzujte, navrhněte a implementujte aplikaci, která bude poskytovat serverovou a klientskou část pro GPSd v nové verzi protokolu (s případnou možností využití staré verze či přímo zasílání NMEA sentencí) prostřednictvím Bluetooth a soketů. U klienta poskytujte data prostřednictvím Mock Location Provider a vizualizujte v něm informace, které není možné dále předat.
4. Výsledné řešení otestujte mezi dvojicí zařízení a vůči oficiálnímu serveru a klientovi a zhodnoťte celkovou funkčnost Vašeho řešení.

Seznam doporučené odborné literatury:

1. Burnette, E. Hello, Android: Introducing Google's Mobile Development Platform. Pragmatic Bookshelf, 2008. ISBN: 978-1-93435-617-3.
2. Meier, R. Professional Android 2 Application Development. Wrox Press, 2010. ISBN: 0-47056-552-0.
3. GPSd - Put your GPS on the net! [online][cit. 2011-10-01]. Dostupné z: <http://www.catb.org/gpsd>.

Formální náležitosti a rozsah bakalářské práce stanoví pokyny pro vypracování zveřejněné na webových stránkách fakulty.

Vedoucí bakalářské práce: **Ing. Pavel Moravec, Ph.D.**

Datum zadání: 18.11.2011
Datum odevzdání: 04.05.2012

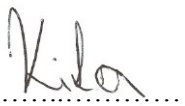
VYSOKÁ ŠKOLA BÁŇSKO-TECHNICKÁ UNIVERZITA OSTRAVA Fakulta elektrotechniky a informatiky

prof. RNDr. Vladimír Vašinek, CSc.
vedoucí katedry

prof. RNDr. Václav Snášel, CSc.
děkan fakulty

Prohlášení studenta

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.



Dne: *1. 5. 2012* ……

Michal Kika

Poděkování

Rád bych poděkoval panu *Ing. Pavlu Moravcovi, Ph.D.* za odbornou pomoc a konzultaci při vytváření této bakalářské práce.

Abstrakt

I v dnešní době existuje stále mnoho zařízení, která neobsahují GPS modul, a proto sdílení dat o poloze je i dnes velice aktuálním tématem.

Hlavním cílem této bakalářské práce je analyzovat, navrhnout a implementovat mobilní aplikaci pro operační systém Android, která bude poskytovat, jak serverovou, tak klientskou část potřebnou ke sdílení informací o poloze. Pro tento účel je využívána nejnovější verze protokolu, navrženého pro projekt GPSd (GPSd-NG). Spojení mezi zařízeními bude realizováno pomocí rozhraní Bluetooth nebo počítačové sítě, tím docílíme širších možností sdílení mezi různými typy zařízení, protože ne všechny zařízení se mohou připojit pomocí počítačové sítě nebo rozhraní Bluetooth.

Klíčová slova

Android, GPSd, NMEA, Bluetooth, Socket, Mock Location Provider

Abstract

Even nowadays, there are still many devices which do not have a built-in GPS module, so sharing of the position data is still an actual topic.

The main goal of this bachelor thesis is to analyze, design and implement a mobile application for the Android operating system that will provide both server and the client, which are necessary for the position sharing. For this purpose, the most recent GPSd protocol designed for GPSd-NG has been used. The connection between the devices will be realized using Bluetooth or a computer network can be achieved by wider opportunities for sharing among different types of devices, because not all devices can connect through a computer network or Bluetooth.

Key words

Android, GPSd, NMEA, Bluetooth, Socket, Mock Location Provider

Seznam použitých zkratek

Zkratka	Anglický význam	Český význam
API	Application Programming Interface	Rozhraní pro programování aplikací
ASCII	American Standard Cod efor Information Interchange	Americký standardní kód pro výměnu informací
C/C++	Programming languages	Programovací jazyky
DGPS	Differential GSP	Diferenciální GPS
DVM	Dalvik Virtual Machine	Dalvik virtuální stroj
EIA	Technical standard	Technický standard
GNSS	Global Navigation Satellite System	Celosvětový navigační satelitní systém
GPS	Global Positioning System	Celosvětový polohový systém
GPSd	GPS daemon	GPS služba
JavaME	Java Micro Edition	Java mikro edice
JavaSE	Java Standard Edition	Java standartní edice
NMEA	National Marine Electronics Association	Národní mořské elektronické sdružení
PDA	Personal Digital Assistant	Osobní digitální pomocník
PPS	Precise Positioning Service	Přesná poziční služba
RTCM	Radio Commission for Maritime Services	Rádio komise pro námořní služby
SPS	Standard Positioning Service	Standardní poziční služba
UHF	Ultra-high frequency	Velmi vysoká frekvence
USB	Universal Seriál Bus	Univerzální sériová sběrnice

Obsah

1 Úvod

V dnešní době je velký technologický pokrok ve vývoji mobilních zařízení, ale bohužel některá stále neobsahují GPS modul, díky němuž by mohly zjistit svou aktuální polohu. Díky tomuto nedostatku mobilních zařízení, mohlo vzniknout téma této bakalářské práce.

Cílem této bakalářské práce je navrhnout a implementovat vlastní jednoduchou aplikaci pro operační systém Android, která bude poskytovat majiteli mobilního zařízení možnost sdílet svou polohu s ostatními mobilními zařízeními. Aplikace bude využívat protokolu GPSd-NG ke komunikaci mezi serverovou a klientskou části a připojení bude realizováno pomocí rozhraní Bluetooth či počítačové sítě.

V prvních kapitolách práce je zohledněna konkurence samotného GPSd, jeho obdoby použitelné pro operační systém Android, ale také pro ostatní operační systémy.

V dalších kapitolách popisuji jednotlivé technologie využité při tvorbě aplikace. Seznámíme se s operačním systémem Android, jeho strukturou a jednotlivými využívanými API při implementaci aplikace. Samotná aplikace řeší sdílení polohy, čili nedílnou součástí tohoto celku je GPS a jeho funkčnost. Seznámíme se ze základní strukturou GPS systému a hlavně s jednotlivými NMEA sentencemi, které server využívá pro tvorbu jednotlivých informačních zpráv pro klienta. Hlavním tématem je samotný protokol, který byl vytvořen pro projekt GPSd a je využíván k vzájemné komunikaci mezi serverem a klientem. V této části si také osvětlíme vývoj jednotlivých verzí tohoto protokolu.

Mezi poslední kapitoly bakalářské práce patří analýza, návrh aplikace a také vlastní implementace, kde bych chtěl čtenáři přiblížit, k čemu slouží jednotlivé třídy a metody v nich vytvořené, aby byl schopen bez větších potíží kódu porozumět. V souvislosti s výše uvedenými kroky bylo také provedeno závěrečné testování aplikace, vedoucí k ověření její celkové funkčnosti.

2 Konkurence GPSd

2.1 ExtGPS

Je aplikace pro mobilní zařízení s operačním systémem Symbian, která byla vytvořena Finskou firmou Symarctic Solutions Ltd [8]. Jejím hlavním úkolem je zajistit sdílení aktuální polohy.

ExtGPS využívá vestavěný přijímač GPS mobilního telefonu k zachytávání NMEA sentencí, které následně předává pomocí Bluetooth nebo USB připojení počítači. Počítač následně data zpracuje a zobrazí na mapě aktuální polohu. Jednou z výhod je možnost zvolit, z jakého zdroje bude poloha získávána. Aplikace umožňuje získávat polohu z integrované GPS, ale i z externích zařízení GPS připojených například přes rozhraní Bluetooth. Další možností získání polohy je přes sítové připojení WiFi nebo připojení poskytované operátorem.

Samotná aplikace je velmi pěkně graficky zpracovaná, v přehledném menu můžeme vidět čtyři ikony symbolizující jednotlivé funkce. Po spuštění aplikace jsou ikony jednotlivých funkcí zašedlé, aby bylo zřejmé, že je v režimu, kdy nesdílí žádné informace. Po stisknutí tlačítka start v menu se ikony aplikace rozzáří barvami a začne se vyhledávat, jak aktivní GPS integrovaná v mobilním zařízení, tak typ připojení USB či Bluetooth. Všechny informace se zobrazí u daných ikon, které navíc upozorňují uživatele svou barvou o momentálním stavu. Aplikace používá tři barvy daných ikon, červená, žlutá, zelená. Každá barva signalizuje uživateli jiný stav.

Aplikace je na první pohled přehledně uspořádaná, propracovaná a jednoduchá na ovládání. Samotný uživatel se s ovládáním rychle a snadno seznámí. Není nic, co bych vytkl. Je vidět, že byla vytvořena profesionální firmou se zkušenými programátory a grafiky, kteří jí dovedli k dokonalosti.

Ve své aplikaci bych se chtěl pokusit o vytvoření podobného grafického rozhraní a jednoduchého ovládaní, které by zvládl každý uživatel již při prvním spuštění.

2.2 GpsGate

Aplikace GpsGate, která je využívaná pro sdílení polohy, byla vytvořena Švédskou firmou GpsGate AB. Zkušenější uživatelé jí také mohou používat jako GPS simulátor nebo Logger. Více informací naleznete zde [9].

GpsGate je vytvořeno pro mobilní zařízení s operačním systémem Windows Mobile. Instalace je snadná, po prvním spuštění se na obrazovce otevře wizard, který pomůže nastavit aplikaci. Wizardem nastavíme typ připojení GPS (Bluetooth GPS, integrovaná GPS, Garmin GPS). Aplikace sama zařízení vyhledá a uživatele o nalezení informuje. Dále spolehlivě nastaví způsob sdílení dat s dalšími účastníky (Bluetooth, ActiveSync, atd). Vše je intuitivní a dobře popsáno, uživatel by neměl

mít žádný problém. Po dokončení nastavení se objeví ikona v tray oblasti. Ikona mění barvu v závislosti na aktuálním stavu přes červenou, žlutou až k zelené. Každá barva má svůj vlastní informativní charakter a díky ní může uživatel vědět, v jakém stavu se právě aplikace nachází.

Aplikace má i další zajímavé funkce, díky kterým se stává zajímavější, je schopna například zachytávat a ukládat NMEA sentence do paměti mobilního zařízení nebo simulovat GPS.

Celé grafické rozhraní aplikace je dobře promyšlené a jednoduché pro použití. Každý uživatel by se měl lehce orientovat a to ne jenom díky přehlednosti grafického rozhraní, ale i díky popiskům doprovázejícím různá nastavení. Vývojáři vytvořili zajímavé webové stránky, které vysvětlují a popisují jednotlivá nastavení aplikace od úplného začátku, až po pokročilejší nastavení, s kterými by si uživatel nemusel vědět rady.

Pro svou tvořenou aplikaci bych využil nejspíš myšlenku zachytávaní a ukládaní NMEA sentencí, které by mohly být zachovány pro další použití.

2.3 GPSd4SE

Aplikace napsaná v jazyce javaME pro mobilní zařízení sdílí dostupné informace z integrovaného GPS modulu. Webová prezentace aplikace není moc zdařilá, ale obsahuje všechny potřebné informace k použití [11].

GPSd4SE je přepracovaná a upravená verze dřívější aplikace J2ME GPS daemon. Tyto verze se od sebe neliší po grafické stránce, ale programová stránka se dočkala změn, díky kterým mohla aplikace fungovat na zařízení Sony Ericsson.

Pokud je aplikace nainstalována můžeme z mobilního zařízení přenášet NMEA sentence pomocí Bluetooth nebo internetového připojení na další zařízení, která jsou schopna data zpracovávat. Po spuštění požádá aplikace o přístup ke GPS modulu, pokud je přístup povolen, automaticky aktivuje GPS a začne číst data. Na displeji můžeme vidět dekódovaná data z NMEA sentencí, která mají informativní charakter, také sledujeme počet připojených klientů, jimž data odesíláme. V aplikaci není mnoho nastavení, můžeme pouze aktivovat a deaktivovat Bluetooth nebo internetové spojení, popřípadě změnit port.

V grafické části aplikace bylo při vývoji použito standardních prvků, které jsou ovšem pro tento typ aplikace dostačující a není co měnit.

Při vývoji své aplikace bych nerad docílil stejného vzhledu a nemožnosti detailnějšího nastavení.

2.4 Bluetooth GPS

Aplikace vytvořena k využívání připojeného externího GPS přijímače přes Bluetooth rozhraní k mobilnímu zařízení. Informace o aplikaci byly čerpány z webové prezentace vývojáře [10].

Při spuštění nás aplikace požádá o aktivaci Bluetooth rozhraní a vybrání jíž spárovaného zařízení GPS. Jakmile je zařízení vybráno, tlačítkem start spustíme čtení NMEA sentencí z GPS přijímače. Na hlavní záložce MAIN vidíme již dekódované NMEA informace, které se zobrazují v daných podoknech se svými příslušnými názvy. Další funkčnost aplikace se nachází v záložce STATUS, kde můžeme vidět informace o satelitech, jejich zakreslené rozmístění a intenzitu signálu. Aplikace umí přímo zobrazit i příchozí NMEA sentence, které můžeme vidět po kliknutí na záložku NMEA Logs.

Významnou funkčností aplikace je možnost zobrazení aktuální polohy na mapě, kterou poskytuje poslední záložka MAP. Pokud je povolen přístup k počítačové síti, stáhnou se mapové podklady od Google, do nichž se zakreslí aktuální poloha. Jednou z dalších funkcí je podpora Mock Location Providera, pokud povolíme tuto schopnost v záložce MAIN, mohou následně ostatní aplikace (Google Maps, GPSStatus, RMaps, OruxMaps, Ndrive, atd), které pracují s aktuální polohou tuto polohu zjistit a vyobrazit.

Při prvním pohledu vidíme, že aplikace není nijak graficky vyšperkována. Při tvorbě aplikace jsou použity standardní prvky z nabídky androidích formulářů. Ale nemyslím si, že by grafické prostředí ubralo této aplikaci na použitelnosti a funkčnosti.

Ve svém GPSd klientovi bych pravděpodobně nejvíce uplatil grafické zpracování informací o satelitech a jejich intenzitě signálu.

2.5 Porovnání aplikací

Všechny aplikace, zmíněné výše, spadají do jednoho celku a to, sdílení dat o poloze. Každá aplikace, která byla vytvořená různými zkušenými programátorskými firmami nebo nadšenými jednotlivci, poskytuje uživateli mnoho různých funkčností, díky nimž jsou tyto aplikace denně využívány. Pro názorné zobrazení a posouzení rozdílů mezi jednotlivými funkčnostmi těchto aplikací, ať už se jedná o možnosti sdílení, možnost využití externího GPS modulu nebo zpřístupnění dat pomocí Mock Location Provideru, je vytvořena tabulka níže.

Tabulka 2.1: Porovnání aplikací

	ExtGPS	**GpsGate**	**Bluetooth GPS**	**GPSd4SE**	**Má aplikace**
Platforma	Symbian	Windows Mobile	Android	JavaME	Android
Sdílení přes Bluetooth	Ano	Ano	Ne	Ano	Ano
Sdílení přes PC síť	Ne	Ano	Ne	Ano	Ano
Možnost využití externího GPS	Ano	Ano	Ano	Ne	Ne
Zasílání RAW NMEA	Ano	Ano	Ano	Ano	Ne
Mock Location Provider	Ne	Ne	Ano	Ne	Ano

3 Android

Společnost Android Inc. byla založena v říjnu roku 2003 v Kalifornii. Po dvou letech, v srpnu roku 2005, ji odkoupil Google Inc., který z ní vytvořil svou dceřinou společnost.

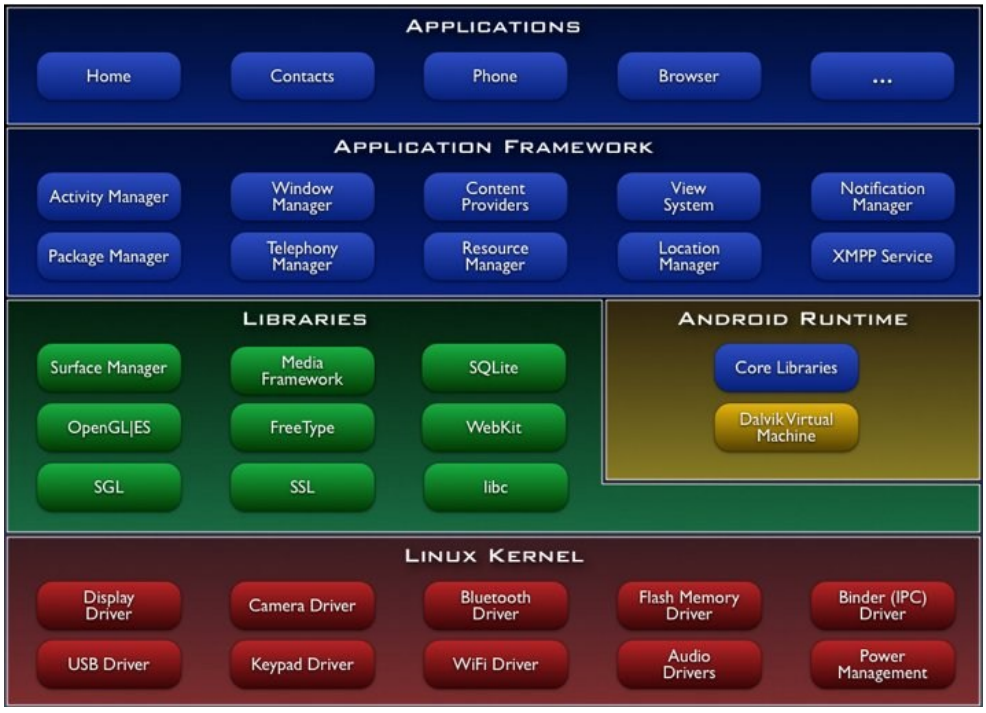
Android je velmi rozsáhlá open source platforma, která byla vytvořena pro využití zejména u mobilních zařízení, například PDA, navigace, tablety, apod. Samotná platforma je vyvíjena konsorciem Open Handset Alliance, jehož hlavním cílem je posilovat, rozšiřovat a urychlovat rozvoj mobilních technologií, které budou mít výrazně nižší náklady na vývoj a distribuci.

Při samotném vývoji systému se kladl důraz na omezení, kterými dostupné mobilní zařízení disponují, například výdrž baterie, málo dostupné paměti, menší výkonnost, apod. Jádro Androidu bylo navrženo tak, aby se podpořil běh na různém hardwaru, systém tak může být využíván například na zařízeních s různou velikostí či různým rozlišením obrazovky.

Více informací o samotném operačním systému Android lze nalézt zde [5], na oficiálních webových stránkách nebo lze použít uvedené publikace [1][2][7].

3.1 Architektura

Architektura operačního systému je rozdělena do 5 samostatných na sebe navazujících vrstev, jak můžeme vidět na obrázku (Obrázek 1).



Obrázek 1: Architektura platformy Android (převzato z http://developer.android.com)

Nejnižší vrstva architektury je jádro operačního systému, které je postaveno na Linuxu ve verzi 2.6 a verzích vyšších. Tato vrstva slouží především ke komunikaci mezi hardwarem a zbytkem softwaru ve vyšších vrstvách.

Nadřazenou vrstvou jsou knihovny, které jsou napsány v jazyce C/C++ a využívají je různé komponenty systému. Knihovny jsou jednotlivým vývojářům poskytovány a zpřístupněny pomocí Android Application Frameworku. Všechny obsažené knihovny můžete nalézt zde [2]. Příklady některých knihoven:

- Media libraries – knihovna umožňující přehrávání video a audio formátů
- SQLite – odlehčená databázová knihovna
- FreeType – knihovna pro vykreslování bitmapových a vektorových fontů
- OpenGL – knihovna pro vykreslování 3D grafiky

Další z vrstev je tzv. Android Runtime, který obsahuje aplikační virtuální stroj Dalvik Virtual Machine (DVM). DVM byl vyvíjen Googlem speciálně pro Android. Jedním z důvodů vývoje byly licenční práva, Java jako jazyk takový a jeho knihovny jsou volně šiřitelné, na rozdíl od JVM. Dalším a hlavním důvodem bylo vytvoření optimalizovaného virtuálního stroje pro mobilní zařízení a to především v oblasti výkonu a úspory energie. Více informací o samotném DVM lze nalézt zde [2]. Android Runtime dále poskytuje základní knihovny programovacího jazyka Java, které se nejvíce přibližují platformě JavaSE.

Nejdůležitější částí zejména pro vývojáře je Aplikační Framework, který poskytuje přístup k velkému počtu služeb, které mohou být využity v jednotlivých aplikacích. Tyto služby mohou zpřístupňovat například prvky uživatelského rozhraní, upozorňovací stavový řádek, hardware používaného zařízení a mnoho dalších funkcí. Mezi služby patří například:

- Prvky View – prvky pro sestavení uživatelského rozhraní, například tlačítka, checkboxy, textové pole, a plno dalších
- Activity manager – řídí životní cyklus aplikací
- Notification manager – umožňuje aplikacím využívat stavový řádek k upozorněním
- Location manager – umožňuje přístup k informacím o poloze zařízení

Poslední nejvyšší vrstva je tvořena vlastními aplikacemi, které využívají uživatelé mobilních zařízení. Tyto aplikace mohou být součástí systému neboli přeinstalované nebo také dodatečně stažené z Obchod Play.

3.2 Android API

Android API tvoří rozhraní pro programování aplikací. Jednotlivé API neboli knihovny obsahují různé metody, funkce nebo třídy, které mohou vývojáři využívat při tvorbě svých aplikací. API určuje, jakým způsobem jsou volány metody knihovny ze zdrojového kódu.

3.2.1 Bluetooth API

Knihovna vývojáři umožnuje přístup a správu Bluetooth rozhraní. Mezi hlavní poskytované funkcionality patří například možnost vyhledávání a párování zařízení, řízení přenosu dat mezi zařízeními, dotazování na spárovaná zařízení, apod.

3.2.2 Location API a Mock Location Provider

Touto knihovnou může vývojář využívat možnosti určování polohy zařízení. Součástí knihovny je mnoho tříd a rozhraní, které vývojáři usnadňují práci s polohou. Jednotlivé třídy a jejich metody nám zpřístupňují například informace o zeměpisné šířce či délce, nadmořské výšce, informace o satelitech, jejich elevaci, azimut, sílu signálu atd. Ovšem knihovna neumožňuje pouze polohu získávat například z GPS přijímače, ale může polohu i vytvářet a předávat jiným aplikacím. Tento systém poskytování polohy se nazývá Mock Location Provider.

4 Globální poziční systém GPS

Je takzvaný celosvětový poziční systém (Global Positioning System). Jeho systém začal vznikat v 60. letech minulého století, kde navázal na původní GNSS Transit projekt a rozšířil ho především svou kvalitou, dostupností a přesností poskytovaných služeb. GPS je především vojenský systém provozovaný ministerstvem obrany Spojených států amerických, s jehož pomocí můžeme určit polohu a přesný čas kdekoliv na Zemi a nad Zemí s přesností do 10 metrů. Přesnost zařízení lze zvýšit až na několik centimetrů, ale tyto služby jsou dostupné pouze pro armádní účely, pro civilní obyvatele je přesnost omezena. V současné době je denní využití systému nedílnou součástí našeho života.

4.1 Struktura systému

Celý poziční systém funguje na jednoduchých principech, které si přiblížíme v následujících kapitolách. Systém můžeme rozdělit do tří základních segmentů, díky kterým lépe pochopíme jeho funkčnost. Více informací o struktuře a samotném GPS systému můžete nastudovat zde [4].

4.1.1 Kosmický segment

Kosmický segment byl původně projektován pro 24 družic, které by obíhaly Zemi po svých oběžných drahách, ovšem dnes je využíván pro 32 družic. Pro další zvyšování počtů družic bude potřebná změna vysílaného signálu, aby se jednotlivé signály nepřehlušovaly. Všechny družice obíhají Zemi ve výšce 20 200 kilometrů, na šesti kruhových drahách se sklonem 55°. Jednotlivé dráhy jsou vzájemně posunuty o 60° a na každé z nich byly původně vypočítané pravidelné pozice pro rozmístění čtyř družic. Nyní z důvodu vyššího počtu družic je na každé dráze pět až šest pozic, které jsou rozmístěné nepravidelně. Jednotlivé družice svou váhou připomínají osobní automobil, váží přibližně 1,8 tuny a na oběžné dráze se pohybují rychlostí 3,8 km/s.

Hlavními částmi družic, které nám umožňují zjistit vlastní polohu a další informace jsou:

- 3 až 4 velmi přesné atomové hodiny s rubidiovým dříve také s cesiovým oscilátorem.
- 12 antén pro vysílání rádiových kódů v pásmu L.
- Antény pro komunikaci s pozemními kontrolními stanicemi v pásmu S.
- Antény pro vzájemnou komunikaci družic v pásmu UHF.
- Optické, rentgenové a pulzní-elektromagnetické detektory, senzory pro detekci startů balistických raket a jaderných výbuchů.
- Solární panely a baterie jako zdroj energie.

V České republice můžeme najednou vidět průměrně osm družic. Minimální počet, který bychom měli přijímačem zachytit je pak šest družic, maximální počet by měl čítat družic dvanáct.

Jednotlivé družice jsou několikrát do roka plánovaně odstaveny pro jejich údržbu. Každá družice projde změnou nastavení atomových hodin a korekcí dráhy, což trvá přibližně 12-24 hodin. Průměrná životnost družic se pohybuje okolo deseti let a celková obměna celého kosmického segmentu trvá přibližně 20 let.

4.1.2 Řídící a kontrolní segment

Je zodpovědný za řízení celého systému. Monitoruje jednotlivé funkce družic a údaje, které získá, předává zpět družicím. Tento řídící segment je tvořen jednou hlavní řídící stanicí, která leží v Colorado Springs, dalšími monitorovacími stanicemi, které tvoří 5 amerických vojenských základen a 3 stanice pro komunikaci s družicemi, které úzce spolupracují s hlavní řídící stanicí.

Hlavním cílem řídícího a kontrolního segmentu je monitorování funkcí jednotlivých družic, sledování a výpočet jejich drah, komunikace a zajištění přesného chodu atomových hodin. Každá závada, která by jednotlivou družici mohla potkat, by se měla co nejrychleji kontrolním segmentem řešit.

Princip fungování je velmi jednoduchý. Pokud družice přeletí nad těmito stanicemi, jsou vyhodnoceny parametry jejich drah a vypočteny korekce, které jsou následně vyslány zpět na jednotlivé družice.

4.1.3 Uživatelský segment

Je tvořen z GPS přijímačů, uživatelů, vyhodnocovacích nástrojů a postupů. Uživatelé pomocí přijímače mohou získávat signály z jednotlivých družic, které jsou momentálně nad obzorem viditelné. Na základě přijatých dat z jednotlivých družic pak přijímač vypočítává polohu, nadmořskou výšku, přesné datum a čas. Komunikace probíhá pouze směrem od jednotlivých družic k uživateli, nikoliv opačně, čili GPS přijímač je pasivní člen. GPS přijímač dále pak komunikuje s počítačem (či jiným zařízením) nejčastěji pomocí protokolu NMEA (National Marine Electronics Association), o kterém se více dozvíme v dalších kapitolách.

GPS přijímače můžeme rozdělit do následujících skupin:

- Podle přijímaných pásem
    - Jednofrekvenční
    - Dvoufrekvenční
    - Vícefrekvenční (připravují se pro pásmo L5)

- Podle kanálů
    - Jednokanálové
    - Vícekanálové
- Podle principu výpočtů
    - Kódová
    - Fázová a kódová

Běžně dostupné přijímače pro denní využití civilními složkami obyvatelstva se vyrábí jako jednofrekvenční, vícekanálové a kódové. Obyčejný přijímač GPS signálu se skládá z antény, předzesilovače, procesoru, časové základny a komunikačního rozhraní.

Uživatelé, kteří mohou využívat systém GPS, můžeme rozdělit do dvou skupin. Do první skupiny bychom zařadili autorizované uživatele neboli vojenský sektor Spojených států amerických a vybrané spojenecké armády. Tyto složky mohou využívat službu PPS, díky které mají zaručenou vyšší přesnost systému. Využívají se především v navádění zbraňových systémů, v mapování, dopravě, atd. V druhé skupině jsou zařazeni ostatní obyvatelé (civilní obyvatelstvo), kteří mohou využívat standard SPS, který není tak přesný. Přijímače vyrobené v USA mají přísné bezpečnostní omezení, pokud nejsou splněna, nesmí být přijímače exportovány. Mezi tyto limity patří například omezení výšky do 18 km a rychlosti do 515 m/s. Těmito preventivními opatřeními se USA snaží řešit možnost zneužití systému ve zbraních, například v raketách nebo střelách s plochou dráhou letu. Využití GPS přijímačů mezi civilními obyvateli výrazně roste, mezi typické odvětví využití patří geodezie, geofyzika, turistika, doprava, apod.

4.2 Princip určování polohy

Každá družice na orbitě vysílá informace o své poloze, přesný čas z atomových hodin, dále pak zná přibližnou polohu ostatních družic. GPS přijímač, který má přímou viditelnost na oblohu, využívá pro výpočet polohy časového rozdílu mezi okamžikem vyslání a okamžikem přijmutí dat. Jestliže přijímač takto zpracuje data ze tří družic, dokáže určit takzvanou 2D polohu, která se skládá ze zeměpisné šířky a délky. Pro výpočet nadmořské výšky potřebujeme signál ze čtyř družic, čímž získáme takzvanou 3D polohu. Pro určování polohy existuje jednoduchá úměra, čím více aktivních družic pro určení polohy využíváme, tím více se výpočet zpřesňuje.

K určení co nejpřesnější polohy je nutné, aby i v přijímači byl přesný čas, kterého docílíme poměrně jednodušším zařízením, než jsou atomové hodiny, z důvodu jejich rozměrnosti a vysoké pořizovací ceny. Tento čas se upraví pokaždé při načítání informací o družicích, podle jejich atomových hodin. Pokud by čas mezi přijímačem GPS a družicemi byl rozdílný, ať už jen on jednu tisícinu vteřiny, chyba v určení polohy by byla řádově stovky kilometrů.

4.3 NMEA 0183

Protokol NMEA byl založen v padesátých letech minulého století skupinou obchodníků, kteří se sešli v New Yorku. Jeho hlavním cílem a úkolem bylo sjednotit komunikaci, vytvořit jednotný standard pro výměnu digitálních dat mezi jednotlivými elektronickými zařízeními, jako jsou například sonary, kompasy, GPS přijímače a mnoho dalších typů nástrojů. Z vyjmenovaných zařízení je zřejmé, že NMEA protokol se využívá zejména u útvarů námořních složek například pobřežní stráže, RTCM, apod. Více informací o vývoji a historii zde [12].

NMEA 0183, který vznikal v devadesátých letech minulého století, nahrazuje dřívější standardy NMEA 0180 a NMEA 0182. V námořních aplikacích se od něj pomalu upouští, z důvodu nástupu novějšího NMEA 2000. Tento novější typ ovšem potřebuje inovaci hardwaru dosavadních zařízení, protože využívá novější elektrický standard EIA-422, který starší zařízení s NMEA 0183 a elektrickým standardem EIA-232 nepodporují.

NMEA 0183 standard používá jednoduché ASCII kódování. Data jsou přenášena sériovým komunikačním protokolem, který definuje, jak budou data předána v jednotlivých sentencích z jednoho vysílače k mnoha posluchačům ve stejný čas.

4.3.1 Pravidla aplikační vrstvy protokolu

Každý protokol, který byl, kdy vytvořen sebou nese určitá pravidla, NMEA nevyjímaje. Než se začneme zabývat rozborem jednotlivých sentenci, které v GPSd využíváme, měli bychom se stručně v bodech s těmito pravidly seznámit.

- Každá zpráva začíná znakem dolaru ($).
- Dalších 5 znaků určuje vysílač, 2 znaky pro ID vysílače a 3 pro typ zprávy.
- Všechny ostatní údaje, které následují, jsou odděleny čárkami. Pokud ovšem některé údaje nejsou k dispozici, zobrazí se pro příklad tento zápis 123,,23,45, kde můžeme vidět několik čárek za sebou, které prezentují, že daný údaj chybí.
- Znak, který následuje po datovém poli je hvězda, která toto pole uzavírá.
- Posledním znakem za hvězdou je kontrolní součet, který je prezentován hexadecimálním číslem. Kontrolní součty se vypočítávají pomocí exkluzivního OR ze všech znaků mezi $ a *.

4.3.2 NMEA sentence

NMEA sentence neboli věty jsou určeny k přenosu dat z přijímače k uživateli pomoci určitého rozhraní. Existuje celá řada druhů těchto vět, jejichž data přenášejí k uživateli mnoho informací, můžeme zde zařadit například informace o poloze, času, rychlosti, počtu družic, apod. Z celé této řady sentencí, které bychom mohli využívat, vybereme pouze nejpoužívanější. Tyto sentence v základu využívá i GPSd. V následujících kapitolách si ukážeme příklady jednotlivých využívaných sentencí a následně každou z nich rozebereme. Pokud ovšem budete mít zájem prostudovat další druhy sentencí, které zde nebudou rozebrány, je možné se s nimi seznámit zde [4] [6].

4.3.3 RMC sentence

RMC sentence se využívá především pro přenos dat o poloze. Pro jednoduchost si ukážeme dvě sentence, jednu takovou, kde již byl signál získán a jednu, kde signál ještě získán nebyl. Postupně je názorně rozebereme na následujícím příkladu.

NMEA sentence 4.1: bez signálu

*$GPRMC,235947.000,V,0000.0000,N,00000.0000,E,,,041299,,*1D*

NMEA sentence 4.2: se signálem

*$GPRMC,092204.999,A,4250.5589,S,14718.5084,E,0.00,89.68,211200,,*25*

Tabulka 4.1: NMEA sentence RMC se signálem

Jednotlivá pole	Příklad	Rozšiřující komentář
ID Sentence	$GPRMC	
UTC Čas	092204.999	Hhmmss.ss
Status	A	A – platné, V – neplatné
Zeměpisná šířka	4250.5589	Ddmm.mmmm
N/S Indikátor	S	N – sever, S – jih
Zeměpisná délka	14718.5084	Dddmm.mmmm
E/W Indikátor	E	E – východ, W – západ
Rychlost	0.00	Uzle
Kurz	89.68	Stupně
UTC Datum	211200	Ddmmyy
Magnetická změna		Stupně
Magnetická změna		E – východ, W – západ
Kontrolní součet	*25	

4.3.4 GGA sentence

GGA sentence uživateli poskytuje mnoho dat o jeho poloze, nadmořské výšce, apod., ale také informace například o počtu využívaných družic.

NMEA sentence 4.3: bez signálu

*$GPGGA,235947.000,0000.0000,N,00000.0000,E,0,00,0.0,0.0,M,,,,0000*00*

NMEA sentence 4.4: se signálem

*$GPGGA,092204.999,4250.5589,S,14718.5084,E,1,04,24.4,19.7,M,,,,0000*1F*

Tabulka 4.2: NMEA sentence GGA se signálem

Jednotlivá pole	Příklad	Rozšiřující komentář
ID Sentence	&GPGGA	
UTC Čas	092204.999	Hhmmss.sss
Zeměpisná šířka	4250.5589	Ddmm.mmmm
N/S Indikátor	S	N – sever, S – jih
Zeměpisná délka	14718.5084	Dddmm.mmmm
E/W Indikátor	E	E – východ, W – západ
Stanovená poloha	1	0 – neplatné, 1 – SPS, 2 – DGPS, 3 - PPS
Použité družice	04	Počet použitých družic
HDOP	24.4	Horizontální přesnost
Nadmořská výška	19.7	
Jednotky nadmořské výšky	M	M – metry
Geoid separace		
Jednotky geoid separace		M – metry
Stáří DGPS		Staří DGPS v sekundách
DGPS id	0000	
Kontrolní součet	*1F	

4.3.5 GLL sentence

Tato sentence má za úkol informovat uživatele pouze o jeho poloze.

NMEA sentence 4.5: bez signálu

*$GPGLL,0000.0000,N,00000.0000,E,235947.000,V*2D*

NMEA sentence 4.6: se signálem

*$GPGLL,4250.5589,S,14718.5084,E,092204.999,A*2D*

Tabulka 4.3: NMEA sentence GLL se signálem

Jednotlivá pole	Příklad	Rozšiřující komentář
ID Sentence	$GPGLL	
Zeměpisná šířka	4250.5589	Ddmm.mmmm
N/S Indikátor	S	N – sever, S – jih
Zeměpisná délka	14718.5084	Dddmm.mmmm
E/W Indikátor	E	E – východ, W – západ
UTC Čas	092204.999	Hhmmss.sss
Status	A	A – platná, V – neplatná
Kontrolní součet	*2D	

4.3.6 GSA sentence

Díky této GSA sentenci k nám přichází data, id aktivních družic, ze kterých odebíráme signál pro určení polohy.

NMEA sentence 4.7: bez signálu

*$GPGSA,A,1,,,,,,,,,,,,,0.0,0.0,0.0*30*

NMEA sentence 4.8: se signálem

*$GPGSA,A,3,01,20,19,13,,,,,,,,,40.4,24.4,32.2*0A*

Tabulka 4.4: NMEA sentence GSA se signálem

Jednotlivá pole	Příklad	Rozšiřující komentář
ID Sentence	$GPGSA	
Mód 1	A	A – automatické 2D/3D, M – vynucené 2D/3D
Mód 2	3	1 – bez polohy, 2 – 2D, 3 – 3D
Použitá družice 1	01	Id použité družice pro určení polohy
Použitá družice 2	20	Id použité družíce pro určení polohy
Použitá družice 3	19	Id použité družice pro určení polohy
Použitá družice 4	13	Id použité družice pro určení polohy
Použitá družice 5 – 12		Id použité družice pro určení polohy
PDOP	40.4	Přesnost polohy
HDOP	24.4	Horizontální přesnost
VDOP	32.2	Vertikální přesnost
Kontrolní součet	*0A	

4.3.7 GSV sentence

Sentence, která nám poskytuje informace o jednotlivých družících nad obzorem. Zde pro příklad a představu přidávám i celkovou GSV zprávu.

NMEA sentence 4.9: bez signálu

*$GPGSV,1,1,01,21,00,000,*4B*

NMEA sentence 4.10: se signálem

*$GPGSV,3,1,10,20,78,331,45,01,59,235,47,22,41,069,,13,32,252,45*70*

NMEA sentence 4.11: kompletní

*$GPGSV,3,1,11,03,03,111,00,04,15,270,00,06,01,010,00,13,06,292,00*74*

*$GPGSV,3,2,11,14,25,170,00,16,57,208,39,18,67,296,40,19,40,246,00*74*

*$GPGSV,3,3,11,22,42,067,42,24,14,311,43,27,05,244,00,,,,*4D*

Tabulka 4.5: NMEA sentence GSV se signálem

Jednotlivá pole	Příklad	Rozšiřující komentář
ID Sentence	$GPGSV	
Počet GSV sentencí celkem	3	Rozsah 1 – 3
Pořadí GSV sentence z celku	1	Rozsah 1 – 3
Počet družic	10	
Id družice 1	20	Rozsah 1 – 32
Elevace 1	78	Rozsah 0 – 90 stupňů
Azimut 1	331	Rozsah 0 – 359 stupňů
SNR 1	45	Intenzita signálu 0 – 99 dB
Id družice 2	01	
Elevace 2	59	
Azimut 2	235	
SNR 2	47	
Id družice 3	22	
Elevace 3	41	
Azimut 3	069	
SNR 3		
Id družice 4	13	
Elevace 4	32	
Azimut 4	252	
SNR 4	45	
Kontrolní součet	*70	

4.3.8 VTG sentence

Tato sentence nepředává informace o samotné poloze, ale informuje o ostatních veličinách, jako je například rychlost a kurz.

NMEA sentence 4.12: bez signálu

*$GPVTG,,T,,M,,N,,K*4E*

NMEA sentence 4.13: se signálem

*$GPVTG,89.68,T,,M,0.00,N,0.0,K*5F*

Tabulka 4.6: NMEA sentence VTG se signálem

Jednotlivá pole	Příklad	Rozšiřující komentář
ID Sentence	$GPVTG	
Kurz	89.68	Stupně
Reference	T	T – platnost čísla
Kurz		Stupně
Reference	M	M – magnetické číslo
Rychlost	0.00	
Jednotky rychlosti	N	N – uzle
Rychlost	0.0	
Jednotky rychlosti	K	K – km/h
Kontrolní součet	*5F	

4.3.9 ZDA sentence

ZDA sentence předává svými daty uživateli časové informace v podání data, času a zóny.

NMEA sentence 4.14: se signálem

*$GPZDA,160012.71,11,03,2004,-1,00*7D*

Tabulka 4.7: NMEA sentence ZDA se signálem

Jednotlivá pole	Příklad	Rozšiřující komentář
ID Sentence	$GPZDA	
UTC čas	160012.71	Hhmmss.ss
Den	11	
Měsíc	03	
Rok	2004	
Zóna	-1	Hodiny
Zóna	00	Minuty
Kontrolní součet	*7D	

5 GPSd

Tento projekt má počátky v roce 1995. Jeho hlavním cílem bylo umožnit uživatelům, kteří vlastní mobilní zařízení bez GPS modulu přijímat informace o poloze ze zařízení, které modul GPS obsahují. GPSd je tedy služba vytvořená na pozadí jakéhokoliv zařízení, pokud je pro daný typ implementována, například mobilní zařízení se systémem Android, různá PC, atd. Pokud má služba běžící na pozadí systému přístupný GPS modul, může získávat informace o poloze, rychlosti, kurzu, atd. Jestliže se ke službě, která čeká na připojení klientů přes počítačovou síť na výchozím portu 2947, připojí klient a vyšle žádost o poskytnutí informací, server používající novou verzi protokolu, začíná klientovi ve formátu JSON předávat zprávy o poloze, které klient zpracuje a vyobrazí nebo poskytne dále ostatním aplikacím. GPSd má svůj vlastní konverzační protokol mezi klientskou a serverovou částí. Tento protokol je důležitý při vývoji a vlastní implementaci aplikace, a proto se v následujících kapitolách s tímto protokolem a jeho vývojem seznámíme. Zájemci o více informací o těchto protokolech je mohou nastudovat zde [13].

5.1 Vývoj protokolu GPSd

Jakákoliv aplikace, protokol nebo jiná komponenta se s určitým časovým úsekem vyvíjí. Takto je tomu i u GPSd, jeho protokoly se vyvinuly až do třetí a zatím nejjednodušší verze. V následujících kapitolách bych chtěl poukázat na vývoj protokolu, kterým mezi sebou server a klient komunikuje.

5.1.1 První protokol

První verze protokolu vznikla v polovině 90. let, byla navržena, jako jednoduchý protokol požadavků a odpovědí. Pokud chceme získat například zeměpisnou šířku a délku, konverzace by vypadala takto:

```
-> P
<- GPSD,P=4002.1207 07531.2540
```

Tyto informace jsou nedostačující, neobsahují čas, odchylky, nadmořskou výšku a spoustu dalších informací. Je to pouze jednoduchý konverzační protokol, který měl minimalizovat využití přenosového pásma.

5.1.2 Druhý protokol

Druhý protokol vznikl na základě požadavků stávajících uživatelů. Uživatelé chtěli mít možnost data vyzvedávat kdykoliv, bylo mnohem výhodnější říci serveru „Mluv!“ a začít data získávat. Tento systém byl implementován a nazván „watcher mode“. Implementace vyžadovala

přidání dvou příkazů do stávajícího protokolu. Pro zahájení a ukončení watcher modu byl vybrán příkaz reprezentován písmenem W. Pro představu, jak tento protokol vypadá, uvádím jednoduchý příklad.

```
-> W=1
<- GPSD,W=1
<- GPSD,O=MID2 1118327700.280 0.005 46.498339529 7.567392712
   1342.392 36.000 32.321 10.3787 0.091 -0.085 ? 38.66 ? 3
<- GPSD,O=MID2 1118327702.280 0.005 46.498345996 7.567394427
   1341.710 36.000 32.321 10.3787 0.091 -0.085 ? 38.64 ? 3
<- GPSD,O=MID2 1118327703.280 0.005 46.498346855 7.567381517
   1341.619 48.000 32.321 10.3787 0.091 -0.085 ? 50.69 ? 3
<- GPSD,Y=MID4 1118327704.280 8:23 6 84 0 0:28 7 160 0 0:8 66 189 45
   1:29 13 273 0 0:10 51 304 0 0:4 15 199 34 1:2 34 241 41 1:27 71
   76 42 1:
<- GPSD,O=MID2 1118327704.280 0.005 46.498346855 7.567381517
   1341.619 48.000 32.321 10.3787 0.091 -0.085 ? ? ? 3
-> W=0
<- GPSD,W=0
```

Zahájení watcher modu se provede pomocí W=1, server následně odpoví a začne vysílat data. Věty označeny znakem O, poskytují informace o poloze, rychlosti, času, atd. Další věta označena znakem Y poskytují informace o satelitech. Další informace o větách poskytujících serverem si můžete nastudovat v technické dokumentaci zde [13].

5.1.3 GPSd-NG protokol

Poslední protokol vznikal mezi lety 2006 až 2009. Hlavní myšlenka autora byla vytvořit jeden objekt, který by obsahoval všechny argumenty. K tomuto účelu byl využit standart JSON. V následující ukázce je znázorněn ekvivalent zprávy O z předešlého protokolu.

```
{"class":"TPV","tag":"MID50","device":"/dev/pts/1",
   "time":"2005-06-09T14:35:11.79",
   "ept":0.005,"lat":46.498333338,"lon":7.567392712,"alt":1341.667,
   "eph":48.000,"epv":32.321,"track":60.9597,"speed":0.161,
   "climb":-0.074,"eps":50.73,"mode":3}
```

Z následující zprávy získáme mnoho informací, například čas, rychlost, stoupání, odchylky, zeměpisnou šířku, zeměpisnou délku a mnoho dalších. Přehled jednotlivých JSON zpráv vytvářených serverem a jejich popis nalezneme zde [3]. Pro vyžádání těchto zpráv ze serveru byl vytvořen příkaz, který po odeslání na server data zpřístupní.

```
?WATCH={"enable":true,"json":true}
```

Zpřístupnění dat příkazem WATCH ve formátu JSON. Příkaz WATCH obsahuje mnoho dalších parametrů, které je server schopen zpracovat a splnit. Před odpojením klienta je vhodné vždy ukončit sledování, aby se příliš nezatěžoval server posíláním dat někam, kde to již není třeba.

6 Analýza a návrh

Po nastudování problematiky GPSd, bylo jasné, že návrh bude rozdělen na dvě části a to část klientskou a serverovou. Tyto dvě části mezi sebou budou komunikovat přes novou verzi protokolu GPSd-NG. Spojení mezi klientem a serverem bude navazováno pomocí počítačové sítě nebo Bluetooth. V následujících podkapitolách, pro přehlednost, rozebereme jednotlivé části klientskou a serverovou odděleně.

6.1 Analýza a návrh serveru

Jednotlivé požadavky kladené na serverovou část, byly zohledněny při následné implementaci aplikace nebo budou později implementovány v dalších verzích aplikace. Požadavků samotných na serverovou část aplikace nebylo mnoho, mezi požadavky patří například využití nového komunikačního protokolu čili GPSd-NG pro komunikaci, možnost využití staršího protokolu vytvořeného pro komunikaci mezi GPSd klientem a serverem, možnost zasílání klientovi přímo jednotlivé NMEA sentence a podpora komunikace, jak přes počítačovou síť, tak i přes Bluetooth rozhraní.

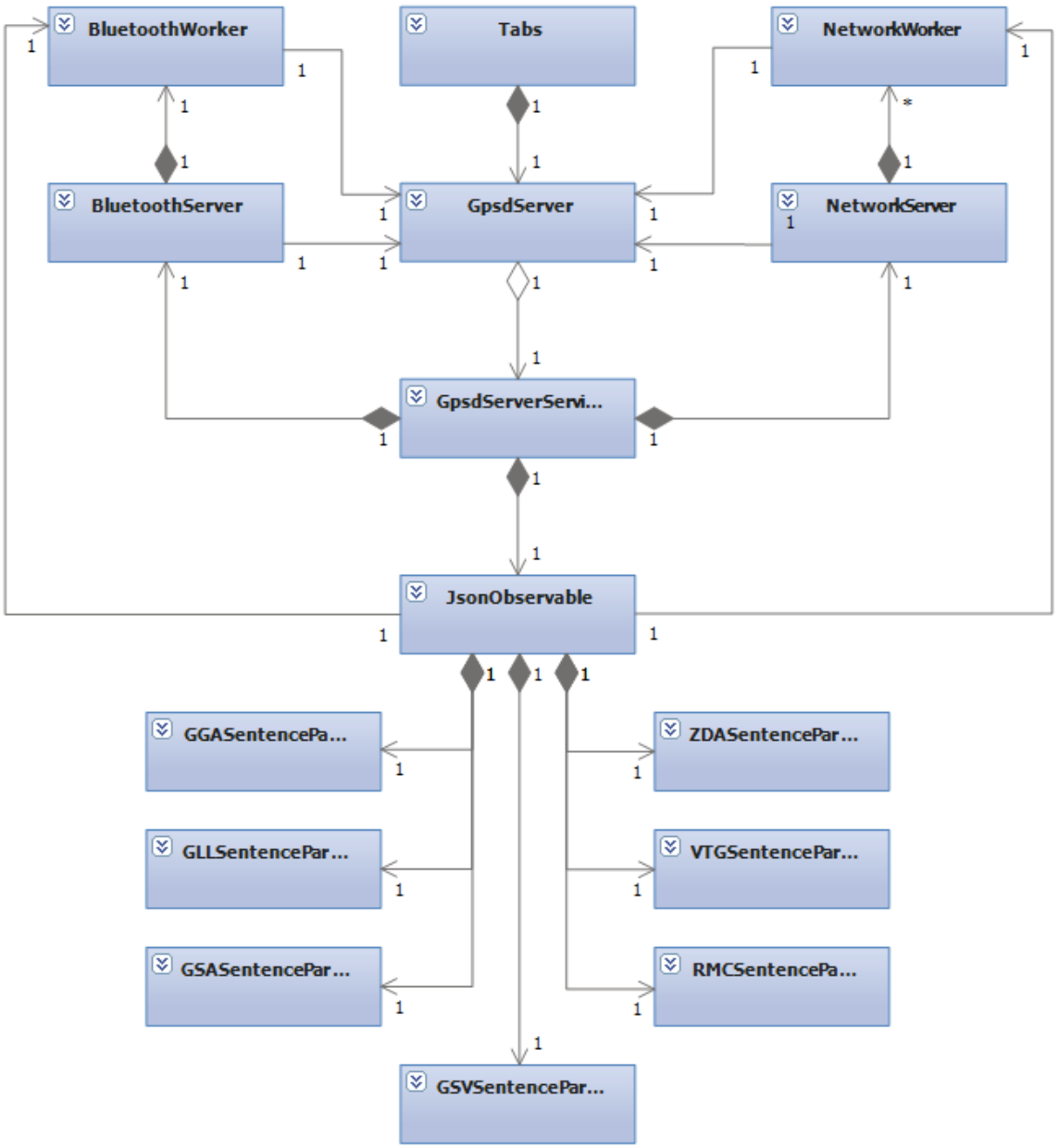
Hlavní jádro aplikace tvoří server, který čeká na příchozí připojení klienta. Samotný server používá Bluetooth rozhraní nebo počítačové sítě. K vytvoření serveru, který využívá ke spojení Bluetooth rozhraní, potřebujeme mít přístup k Bluetooth API, který je součástí systému Android a poskytuje funkcionality k jeho ovládání. Pro vytvoření serveru, který využívá ke komunikaci počítačovou síť, musíme použít Net API, který poskytuje dané třídy s jednotlivými metodami. Server by měl podporovat více připojených klientů. Aby byl server schopen komunikovat s více než jedním klientem, bylo potřeba zvolit multi-klientskou komunikaci, která se v Javě běžně realizuje multi-vláknovým serverem.

Jednou z hlavních úloh serveru je dekódovat NMEA sentence a získávat z nich informace. Existuje mnoho knihoven, které jsou volně ke stažení a připraveny k využití v aplikacích, ale všechny tyto knihovny jsou pouze beta-verze nebo nesplňují mé požadavky. Abych měl vše pod kontrolou a nemusel se spoléhat na jiné knihovny, zvolil jsem nakonec k dekódování sentencí vlastní implementaci dekodéru. K získávání jednotlivých NMEA sentencí je zapotřebí mít přístup k samotnému GPS modulu, ten je v systému Android zprostředkováván pomocí Location API, který poskytuje dané třídy, metody a hlavně posluchače pro příjem NMEA sentencí. Jednotlivé přehledné ukázky kódu i postup, jak s Location API pracovat, můžete najít zde [1] nebo v již odbornějším zde [2].

Další úvaha spočívala nad tím, jak nejlépe uskutečňovat spojení mezi jednotlivými vlákny a třídou, která bude vytvářet a poskytovat informace pro klienta ve formátu JSON zpráv. První úvahou byla možnost vytvořit klasického posluchače, který bude informovat jednotlivá vlákna o příchodu nových zpráv a následně je i předávat. Tato konstrukce se nakonec jevila zbytečně komplikovaná, a proto jsem zvolil jednodušší řešení, které je již v Javě implementováno. Daným řešením byl Observer, který poskytuje jednoduchou funkčnost registrace posluchačů a jejich následné seznamování s průběžnými změnami dat. Toto řešení bylo zvoleno z důvodu jeho jednoduchosti a z nepotřebnosti implementovat další rozsáhlé kódy, jako u klasického posluchače.

Hlavním důvodem vývoje této aplikace je použít protokol GPSd, který byl pro tuto skutečnost sdílení informací o poloze vyvinut. GPSd obsahuje několik verzí protokolů a mým úkolem bylo implementovat využívání nejnovější verze s názvem GPSd-NG, popřípadě verze starší nebo poskytovat RAW NMEA. Ovšem při nastudování technické dokumentace k jednotlivým protokolům jsem se rozhodl, že nejlepší cestou bude implementovat pouze nejnovější protokol, důvodem byla jeho celková jednoduchost a propracovanost. Komunikace pomocí starší verze protokolu nebo zasílání jednotlivých NMEA sentencí bude zohledněna v dalším vývoji aplikace.

Při návrhu uživatelského rozhraní bylo dbáno na jeho přehlednost a snadnou ovladatelnost. Uživatelské rozhraní jsem rozvrhl na jednu hlavní část, kterou tvoří dvě záložky. První záložka bude obsahovat klientskou část a druhá serverovou. Serverová část dále bude obsahovat hlavní okno s možnosti spuštění serveru a okno s informativními zprávami pro uživatele. Po zobrazení menu se budou moct uživatelé přesunout do okna nastavení, kde lze zvolit port a typ serveru. Pro účely testování a informovanosti uživatele jsem zařadil i okno s vyobrazením přijatých informací od klientů a odeslaných informací serverem jednotlivým klientům.



Obrázek 2: UML třídní diagram serverové části

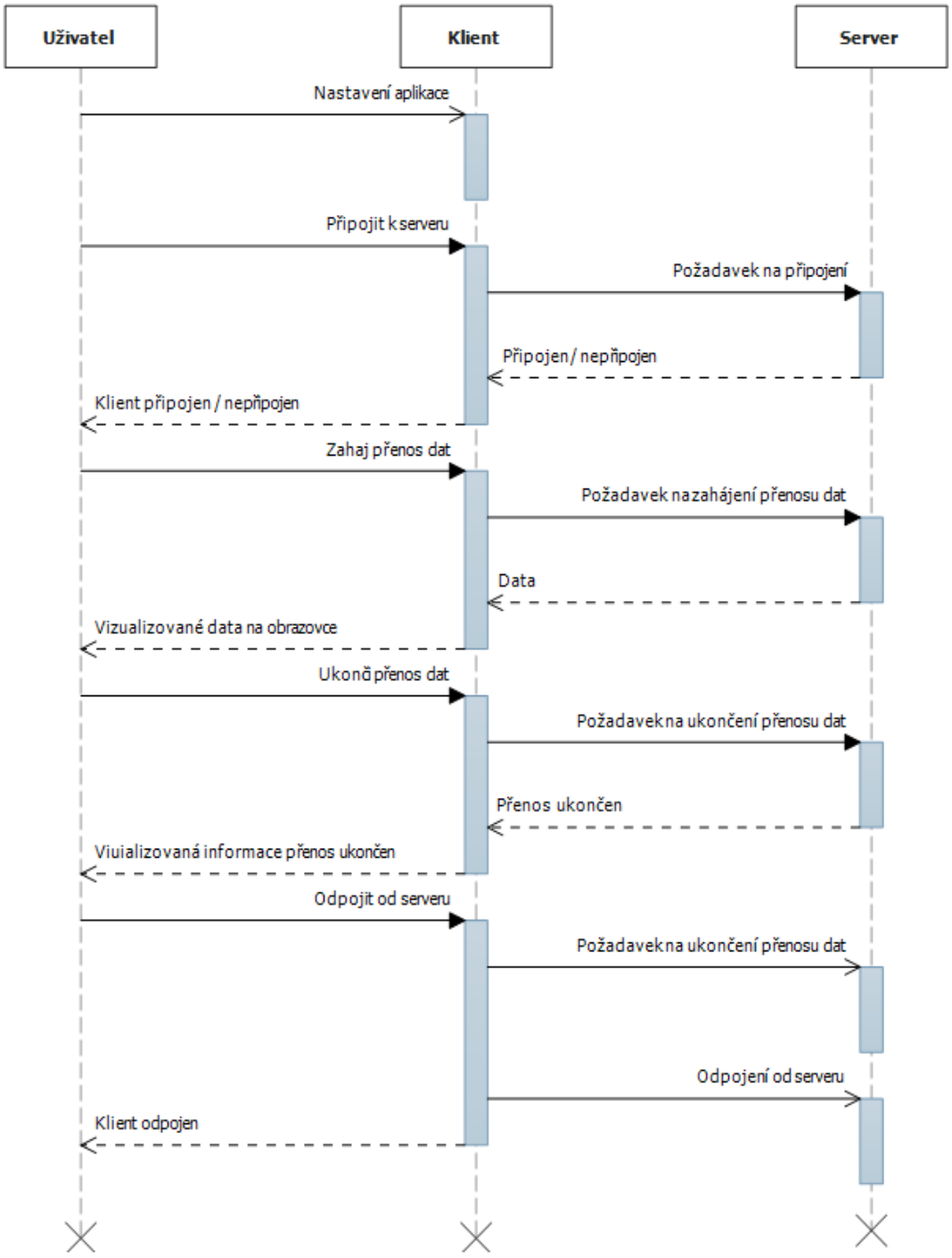
6.2 Analýza a návrh klienta

Klientská část byla pro návrh nejsnadnější, protože požadavků na ni nebylo mnoho. Mezi základní požadavky patří například vizualizace získaných informací ze serveru a možnost poskytnout informace dále, například ostatním aplikacím, vytvořením Mock Location Providera.

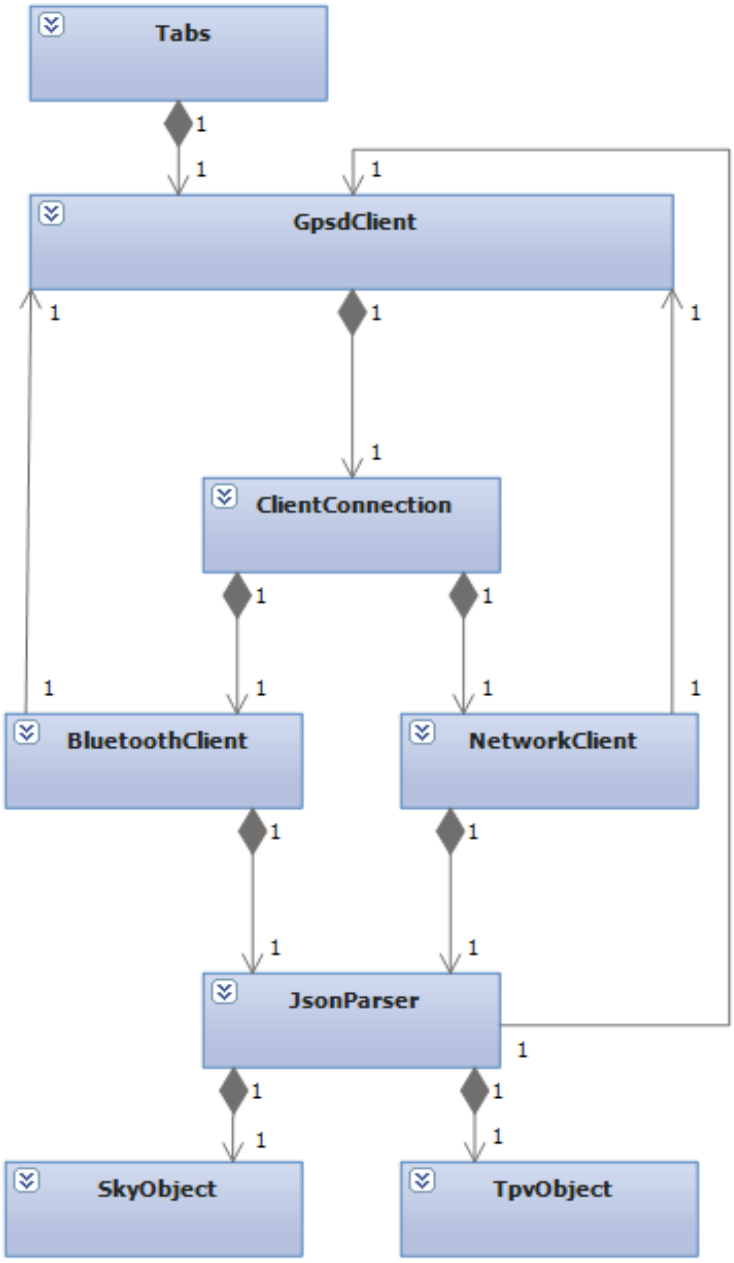
Samotnou aplikaci jsem navrhoval od konce, čili od připojení k serveru, které je realizováno pomocí Bluetooth rozhraní nebo počítačové sítě. Pro připojení k serveru pomocí počítačové sítě je nutné využít v systému Android Net API, které poskytuje třídy pro vytvoření a práci s připojením. Ke komunikaci se serverem, jak je zmíněno výše, je možné použít i rozhraní Bluetooth, které reprezentuje v systému Android Bluetooth API obsahující všechny potřebné třídy a metody pro naši práci. Jednotlivá připojení na server se budou uskutečňovat pomocí samostatného vlákna, aby se nezatěžovalo hlavní vlákno uživatelského rozhraní a nezamrzalo.

Dalším krokem návrhu bylo zvolit vhodný způsob dekódování příchozích dat ze serveru a jejich následné vizualizace. V tomto případě mě napadl pouze jeden způsob. Myšlenka byla následující a to vytvořit třídy, které budou reprezentovat jednotlivé příchozí objekty TPV a SKY, popřípadě další, dekódovaná data do nich vložit a následně předat celý objekt k vizualizaci. Jednotlivé JSON objekty ovšem nemusí obsahovat stále všechny parametry, tak jsem hledal způsob, jak jednoduše tento problém odstranit. Nabídla se mi metoda obsažena v samotném dekodéru JSON objektů v Javě, která je schopna detekovat přítomnost dané proměnné v JSON objektu, pokud přítomna je, její hodnotu získá, ale jestliže v objektu není, vrátí prázdný řetězec.

Samotné uživatelské rozhraní bude začínat jako u serverové části, přece jenom jde o jednu ucelenou aplikací rozdělenou dvěma záložkami do dvou samostatných oddělených částí. Záložka klienta nám bude nabízet hlavní obrazovku s možností připojení na server, s možností zapnutí sledování, čili vyžádání dat ze serveru, dále také průběh komunikace mezi serverem a klientem. Průběh komunikace mezi klientem a serverem s jednotlivými uživatelskými zásahy na klientské části můžeme prezentovat následujícím obrázkem (Obrázek 3). Pro klienta je navrhnuto přehledné menu, jež uživateli bude umožňovat přecházet mezi jednotlivými okny aplikace. Další okno uživateli umožní měnit jednotlivá nastavení aplikace, jako je například IP adresa serveru, jeho port nebo základní zvolení typu připojení k serveru. Poslední položka v menu zpřístupní uživateli již přijaté informace ze serveru a přehledně je vizualizuje.



Obrázek 3:Aktivitní diagram – komunikace klienta se serverem



Obrázek 4:UML třídní diagram klientské části

7 Implementace

V této kapitole bych se chtěl věnovat vlastní implementaci aplikace, rozebrat implementované metody jednotlivých tříd a jejich funkčnost. Samotná aplikace byla vyvíjena v prostředí Eclipse IDE pod operačním systémem Linux. Linux, byl zvolen pro možnost testování vyvíjené aplikace, protože GPSd podporuje pouze tento systém. Pro samotný vývoj aplikace a jako minimální verzi jsem zvolil Android 2.3.3. s pomocnými Google APIs level 10, protože je nejrozšířenější verzí na trhu s mobilními zařízeními a obsahuje ji také reálné testovací mobilní zařízení. Chtěl jsem mít možnost aplikaci reálně otestovat a nepoužívat pouze emulátor, který neposkytuje všechny funkčnosti. V dalším vývoji aplikace chci ovšem jednotlivé verze Androidu zohlednit a rozšířit tak podporu o další verze systému.

7.1 Rozbor jednotlivých tříd serveru

Serverová část se zpočátku jevila jako jednoduchá záležitost, ale s postupným odkrýváním jednotlivých funkčností a začátečnickou znalostí vývojového prostředí a Android SDK se implementace značně ztížila.

7.1.1 Tabs

Tato třída, která dědí z TabActivity vytváří rozdělení obrazovky na dvě záložky. Jedna záložka obsahuje aktivitu serveru, kterou vyvolává pomocí třídy Intent a druhá záložka obsahuje aktivitu klienta. Tato třída je výchozí při spouštění celé aplikace, proto je při spouštění kontrolováno, jestli je aplikace spuštěna pomocí ikony v menu nebo pomocí kliknutí na notifikační oblast. Pokud je aplikace spuštěna z notifikační oblasti je zřejmé, že uživatel chce manipulovat se serverovou částí. Přepne se proto ze standardního nastavení, kde se prvně zobrazí záložka klienta, na nastavení, kde se zobrazí prvně záložka serveru.

7.1.2 GpsdServer

Implementuje uživatelské rozhraní serveru s tlačítky, nastavením, logem a menu. Hlavním účelem této třídy je umožnit uživateli nastavit aplikaci, nastavení ukládat a spustit samotnou službu. Se službou může třída nadále komunikovat i po opětovném spuštění, vyobrazovat jednotlivé informace o připojení, odpojení, spuštění či ukončení serveru. Dále pak umožňuje pomocí menu přepínat mezi jednotlivými okny aplikace.

Implementované metody

- onCreate
  - Inicializuje jednotlivé prvky uživatelského rozhraní.
  - Kontroluje, zda je služba spuštěna a pokud ano, musí se připojit.
  - Při spuštění načítá uložené nastavení uživatele.
- onCreateOptionsMenu
  - Vytváří menu pro danou aktivitu.
- onOptionsItemSelected
  - Detekuje uživatelem zvolenou položku v menu a vykoná danou operaci.
- onStop
  - V této metodě se při stopnutí aplikace uloží uživatelská data a odpojí se od služby.
- isMyServiceRunning
  - Metoda, která testuje, zda služba již běží nebo ne.
- btnServerSocket
  - Obslužná metoda tlačítka zapnutí a vypnutí serveru.
- isBluetoothEnable
  - Kontroluje, zda je zapnutý Bluetooth.
- showAlertDialog
  - Dialog, který informuje uživatele v případě že Bluetooth není aktivní.
  - Je vyvolán metodou isBluetoothEnable.
- onTouch
  - Umožnuje uživateli pohybovat se mezi jednotlivými okny pomoci dotyku.
- handleMessage
  - Metoda čeká na příchozí zprávy od služby a upravuje uživatelské rozhraní.
- onServiceConnected
  - Metoda je vyvolána po připojení ke službě a registruje posluchače pro příjem zpráv.
- onServiceDisconnected
  - Metoda je vyvolána, pokud je služba ukončena mimo aplikaci.
- doBindService
  - Metoda pro připojení aplikace ke spuštěné službě.
- doUnbindService
  - Při ukončování aplikace je metoda zavolána a ukončí komunikaci se službou.

7.1.3 GpsdServerService

Tato třída implementuje službu, která pracuje na pozadí bez uživatelského rozhraní [2]. Základním prvkem služby je vytvoření notifikace, která uživatele informuje o spuštění. Samotná třída při svém startu spustí vlákno serveru podle parametrů, které byly předány službě při jejím vytváření. Pokud se uživatelské rozhraní uzavře, služba dále shromažďuje veškeré informace o připojených klientech a po opětovném připojení samotné aplikace tyto informace předá.

Implementované metody

- handleMessage
  - Lokální metoda pro příjem zpráv z podřízených vláken.
- handleMessage
  - Metoda vytvořena pro příjem zpráv z uživatelského rozhraní.
- onBind
  - Po připojení aplikace jí předá vytvořenou instanci messengra pro možnost komunikace mezi službou a aplikací.
- onDestroy
  - Vyvoláno při ukončení služby.
  - Ukončuje server, zruší notifikaci a čtení z GPS.
- onCreate
  - Při vytvoření služby připraví notifikaci ke spuštění.
- onStart
  - Při zapnutí služby se spustí tato metoda, která spustí podle přijatého nastavení server.
- notificationStart
  - Metoda spouští notifikaci.
- notificationStop
  - Metoda ukončí notifikaci z oznamovací oblasti.
- startBluetoothServer
  - Vytvoří instanci třídy Bluetooth serveru a spustí jí ve vlastním vlákně.
- startNetworkServer
  - Vytvoří instanci třídy síťového serveru a spustí jí ve vlastním vlákně.

7.1.4 BluetoothServer a NetworkServer

Jednoduché třídy, které implementují funkčnost Bluetooth a síťového serveru a čekají na připojení jednotlivých klientů.

Implementované metody

- BluetoothServer a NetworkServer
  - Konstruktor, který při vytváření instance třídy přejímá řadu informací.
- run
  - Implementovaná metoda vlákna, která po spuštění čeká na připojení klienta.
- isStopped
  - Metoda ke zjištění, zda je server stopnut.
- stop
  - Metoda, která uzavře BluetoothSocket a uzavře komunikaci se všemi klienty.

7.1.5 BluetoothWorker a NetworkWorker

Třídy, které přímo implementují komunikaci s klientem. Čekají na příchozí požádání klienta o příjem dat.

Implementované metody

- BluetoothWorker a NetworkWorker
  - Konstruktor, který přejímá příchozí socket a další parametry, s kterými třída dále pracuje.
- run
  - Metoda vlákna, která pracuje s příchozími daty od klientů.
- update
  - Tato metoda je implementována z rozhraní Observer.
  - Metoda přijímá data z JsonObservable a poskytuje je klientovi.
- stop
  - Ukončí komunikaci.
- sendVersion
  - Pošle klientovi informace o verzi.
- sendDevice
  - Pošle klientovi informace o zařízení.
- sendWatch
  - Pošle klientovi zpětnou reakci na příkaz WATCH.

7.1.6 GGA, GLL, GSA, GSV, RMC, VTG, ZDA SentenceParser

Tyto třídy implementují dekodér pro jednotlivé NMEA sentence. Hlavním cílem bylo vytvořit co nejefektivnější a nenáročný dekodér, který by mohl fungovat na většině zařízení.

Implementované metody

- xxxSentenceParser
  - Konstruktor, kterému předáme danou sentenci.
- isValid
  - Kontroluje, zda je sentence validní.
- getChecksum
  - Vypočítá kontrolní součet z NMEA sentence.
- parse
  - Dekóduje jednotlivé sentence.

7.1.7 JsonObservable

Tato třída je velmi důležitá pro celou serverovou část. Přijímá totiž jednotlivé NMEA sentence z GPS modulu a dekóduje je pomocí jednotlivých dekodérů uvedených níže. Po dekódování NMEA sentencí vytvoří JSON objekty a rozešle je mezi jednotlivé zaregistrované Bluetooth a network workery, kterým zpráva přichází pomocí implementované metody update.

Implementované metody

- JsonObservable
  - Konstruktor.
- stop
  - Ukončí sledování GPS.
- stopWorkers
  - Řekne jednotlivým vláknům, které komunikují s klienty, aby se ukončily.
- onNmeaReceived
  - Automaticky implementována metoda pro příjem z NMEA zpráv.
- tpv
  - Metoda vytvoří tpv objekt z přijatých dat.
- sky
  - Metoda vytvoří sky objekt z přijatých dat.
- satellite
  - Metoda je vyvolána metodou sky.
  - Vrací souhrn informací o satelitech.
- calcDateTime
  - Metoda vypočítá čas v milisekundách.

7.1.8 SatelliteInfo

Pomocná třída vytvořená k dekódování GSV vět. Bylo zapotřebí uchovat data o satelitech jejich azimut, elevaci, atd. Třída obsahuje pouze proměnné a jejich přidružené metody get a set.

7.2 Rozbor jednotlivých tříd klienta

7.2.1 GpsdClient

Implementuje rozhraní klienta pro jednotlivé uživatele. Poskytuje možnosti nastavení a vizualizuje získané informace přijaté ze strany serveru.

Implementované metody

- onCreate
  - Při vytvoření aktivity inicializuje komponenty a načte uložená data uživatelem.
- onCreateOptionsMenu
  - Vytvoří menu pro danou aktivitu.
- onOptionsItemSelected
  - Čeká na zvolení položky v menu a vykoná danou operaci.
- setProperties
  - Metoda je volána z onCreate, byla vytvořena pro přehlednost a obsahuje inicializace uživatelského rozhraní.
- handler
  - V této metodě vytvářím rozhraní pro komunikaci s podřízenými vlákny.
- onStop
  - Při ukončování aplikace se uloží uživatelská nastavení.
- onDestroy
  - Metoda ukončuje připojení na server a zruší Mock Location Providera.
- startMockLocation
  - Metoda inicializující Mock Location Providera.
- stopMockLocation
  - Metoda, která odstraní Mock Location Providera.
- connect
  - Obslužná metoda tlačítka připojit k serveru.
- watch
  - Obslužná metoda tlačítka sledování pro žádost o data.
- createSatInfo
  - Metoda dynamicky vizualizuje informace o satelitech podle jejich počtu.
- isBluetoothEnable
  - Kontroluje, zda je Bluetooth aktivní.
- showAlertDialog
  - Dialog je vyvolán metodou isBluetoothEnable pokud Bluetooth není aktivní.
- getBluetoothDevices
  - Zjišťuje spárovaná zařízení a vizualizuje je uživateli, aby si mohl vybrat.
- onItemClick
  - Obslužná metoda seznamu spárovaných zařízení, která čeká na výběr.

7.2.2 ClientConnection

Tato třída vznikla za účelem zjednodušení. Vytváří připojení na server pomocí jednotlivých tříd NetworkClient a BluetoothClient. Třídě se předají informace, které uživatel nastavil v aplikaci. Tyto informace obsahují například typ připojení na server popřípadě port a IP adresu serveru. Zjednodušení je v tom, že aktivita GpsdClient nemusí řešit dalším rozsáhlým kódem připojování na server, pouze předá třídě informace a ta se postará o zbytek. V implementovaných metodách zmiňuji pouze hlavní metody, bez obslužných metod typu get a set.

Implementované metody

- start
  - Podle parametrů spustí nové vlákno klienta.
- stop
  - Pošle klientovi zprávu, ať ukončí komunikaci se serverem.
- watchEnable
  - Zahajuje přijímání dat ze serveru.
- watchDisable
  - Ukončuje příjem dat ze serveru.

7.2.3 NetworkClient a BluetoothClient

Tyto třídy byly implementovány, jako hlavní prostředek pro komunikaci se serverem. Jednotlivým třídám, pokud jsou spouštěny, předá parametry třída ClientConnection. Následně se vytvoří spojení se serverem na určitém portu a IP adrese, popřípadě s určitým zařízením spárovaným přes Bluetooth. Uživatelské rozhraní informuje o všech krocích připojení.

Implementované metody

- NetworkClient a BluetoothClient
  - Konstruktory tříd, které přebírají parametry.
- run
  - Obslužná metoda vlákna pro připojení a cyklický příjem dat ze serveru.
- watchEnable
  - Posílá serveru žádosti o příjem dat.
- watchDisable
  - Metoda posílá serveru žádost o ukončení příjmu dat.
- stop
  - Slouží k ukončení komunikace se serverem.

7.2.4 JsonParser

Již z názvu třídy vyplývá, že se jedná o třídu, která implementuje metody pro dekódování příchozích JSON objektů ze strany serveru. Dekódované informace dále předává uživatelskému rozhraní, které je vizualizuje pro uživatele.

Implementované metody

- JsonParser
  - Konstruktor, přes který se předá vazba na uživatelské rozhraní.
- parseJson
  - Samotný dekodér.
  - Rozlišuje TPV a SKY objekty.

7.2.5 SkyObject a TpvObject

Každá z těchto tříd reprezentuje jeden objekt vytvářený serverem. Obsahují pouze get a set metody pro nastavení a následný výběr dat k jejich vizualizaci v uživatelském rozhraní.

7.3 Problémy při implementaci

Hlavním problémem, se kterým jsem bojoval po celou dobu implementace, byla neznalost prostředí a SDK Androidu. S každou novou myšlenkou k implementaci aplikace jsem musel dostudovávat dokumentaci a vyhledávat, jak daný problém vyřešit. Proto vývoj celé aplikace trval poněkud déle, ale na druhou stranu jsem se seznámil se základními programovacími prostředky Androidu.

Dalším problémem, nyní ovšem na serverové části bylo, jak implementovat funkčnost obesílání jednotlivých připojených klientů daty. První idea byla implementovat rozhraní posluchače, který bude do jednotlivých tříd prostupovat a předávat data. Ovšem tato skutečnost se nakonec ukázala nevhodná a složitá na uskutečnění, proto jsem zvolil jednodušší typ předávání dat pomocí Observeru a tříd s ním spojených.

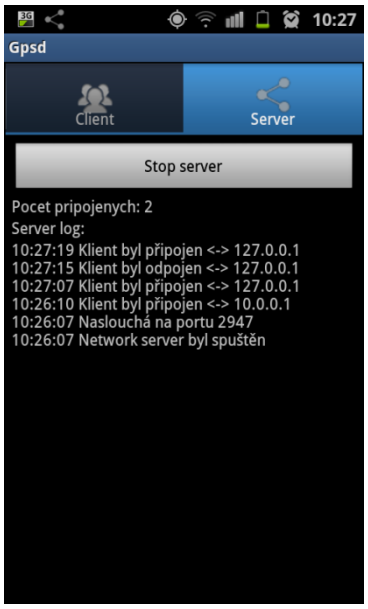
Další problém vznikl při implementaci dekodéru pro jednotlivé NMEA sentence. Problém vyvstal, když sentence měla vynechané některé hodnoty. Dekodér na tuto skutečnost v některých případech nebyl připraven a hlásil chyby. Proto jsem implementoval jednoduchý cyklus, který v první řade, po načtení sentence, nahradí prázdné položky slovem null. Pokud dekodér null detekuje, neuloží data do dané proměnné, ale přeskočí na další položku v sentenci.

8 Testování

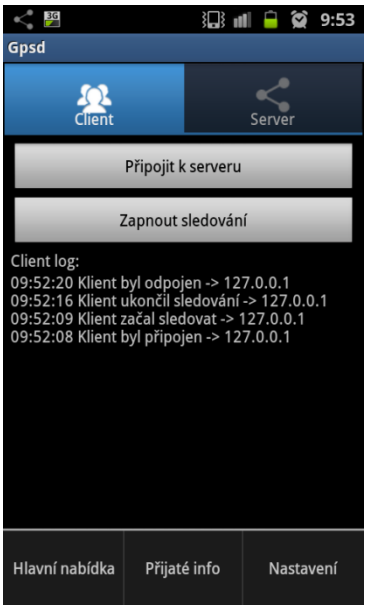
Samotné testování při vývoji aplikace probíhalo v emulátoru a v reálném mobilním zařízení Galaxy SII od firmy Samsung. V případě vývoje serverové části neposkytoval emulátor dostatečnou funkčnost, proto jsem byl nucen aplikaci testovat v reálném zařízení, které obsahuje GPS modul a Bluetooth rozhraní, které emulátor nepodporuje.

8.1 Dvě instance vytvořené aplikace společně

Samotná výsledná aplikace byla testována mezi sebou, a to, jak na jednom zařízení na localhostu, tak i mezi dvěma zařízeními, kde spojení bylo realizováno přes síť a Bluetooth rozhraní.



Obrázek 6: Serverová část aplikace



Obrázek 5: Klientská část aplikace

Na obrázcích (Obrázek 6, Obrázek 5) můžeme vidět testování aplikace. K serverové části byl připojen klient pomocí sítě z jiného mobilního zařízení, které reprezentuje IP adresa 10.0.0.1. Dále byl k serveru připojen klient z localhosta, kterého reprezentuje IP adresa 127.0.0.1. Připojení obou klientů bylo bezproblémové a oba si posléze vyžádali data od serveru. Síťové přenosy můžu tímto hodnotit jako funkční a bezproblémové.

Spojení klientské a serverové části pomocí Bluetooth rozhraní, již nebylo tak dokonalé. Spojení sice proběhlo mezi zařízeními, data byla vyžádána, ale komunikace se serverem není stále bezchybná. Jelikož Bluetooth rozhrani je stále ve stádiu testování počítám s brzkou stoprocentní funkčností.

8.2 Klient s oficiálním serverem

Na oficiálních webových stránkách problematiky GPSd je volně ke stažení balík pro Linux díky kterému je možno vytvořit server. Klient byl testován s verzí balíku 3.5.

Při vytvoření serveru v operačním systému Linux byly použity testovací data, která obsahuje samotný balík. Poté byl nainstalován a spuštěn gpsfake server na localhostu PC.

Samotné testování klienta vůči serveru probíhalo průběžně s vývojem aplikace. Při testování se klient postupně vyvíjel do aktuální podoby, jak lze vidět na obrázku výše (Obrázek 5). Díky testování se našly a postupně odstranily chyby, které by mohly mít v budoucnu na aplikaci nežádoucí vliv. Ovšem nepočítám s tím, že by aplikace byla již bez jakýchkoliv chyb a stoprocentní. Testování probíhalo přes síťové rozhraní, nikoli přes Bluetooth, ten totiž není oficiálním serverem podporován, protože aplikace byla původně vyvíjena pro PC a sdílení přes počítačovou síť. Nynější verze klienta komunikuje s oficiálním serverem bez větších problému.

8.3 Server s oficiálním klientem

Oficiální klient je také součástí balíku pro Linux k potřebám testování. Verze balíku, se kterým byl server testován, je 3.5 jako u testování klienta.

Serverová část vyvinuté aplikace byla spuštěna na reálném mobilním zařízení. Samotná data, která byla při testování použita, byla získávána z integrovaného GPS modulu. Klienta v balíku pro systém Linux reprezentuje aplikace s názvem „xgps“, která se po spuštění s parametry (IP adresa, port) připojí na server a začne okamžitě čerpat data, která se vizualizují v přehledném uživatelském rozhraní.

Testování s oficiálním klientem nebylo u serveru praktikováno při jeho průběžném vývoji, ale ani tato skutečnost nezabránila, aby finální verze serveru s oficiálním klientem nebyly kompatibilní. Spojení bylo navázáno pouze pomocí počítačové sítě, protože klient nepodporuje jiné možnosti připojení. Komunikace probíhala bez sebemenších potíží.

9 Závěr

Cílem této bakalářské práce nebylo pouze implementovat aplikaci, která obsahuje klientskou i serverovou část a využívá ke komunikaci protokolu GPSd-NG, ale také seznámení se s danou problematikou GPSd a ostatními používanými technologiemi při vývoji aplikace.

Při vypracovávání této práce jsem využil své dosavadní znalosti získané při studiu hlavně v oblasti programování v jazyce Java, ale také jsem si rozšířil obzory o programování aplikací pro systém Android, což bylo pro mě velmi přínosné a určitě své nabyté znalosti v budoucnu dále využiji a rozšířím o další poznatky.

Samotná implementace probíhala s velkým zájmem o možnost poznat nové technologie. V průběhu práce jsem se seznámil například s jednotlivýma NMEA sentencemi a jejich využitím v praxi. Dále jsem se setkal s Bluetooth rozhraním, které jsem bohužel nemohl v aplikaci řádně otestovat, protože emulátor Bluetooth nepodporuje. Znalosti jsem si rozšířil také o možnosti implementace vlastního serveru a klienta. Všechny tyto nabyté poznatky jsem se snažil využít při vypracovávání této práce.

Aplikace, jako taková, splňuje základní požadavky, které na ní byly kladeny. Tím nejzákladnějším požadavkem bylo využívání ke komunikaci mezi serverem a klientem protokolu GPSd-NG, což bylo splněno, ale protokol je velmi rozsáhlý a poskytuje velmi mnoho možností, ze kterých bylo využito velmi málo a v budoucnu při rozvoji aplikace bych chtěl tento fakt změnit a rozšířit aplikaci o plnou podporu protokolu popřípadě i protokolů starších verzí. Aplikace podporuje komunikaci přes TCP protokol, který funguje velice dobře, jak na serverové, tak i na klientské části. Bluetooth rozhraní využívané aplikací je na tom o poznání hůře, je to způsobeno tím, jak je již zmíněno v předešlých kapitolách, že jsem nemohl řádně tuto problematiku otestovat. Dalším požadavkem na aplikaci bylo umožnit klientovi přijatá data ze serveru, prostřednictvím Mock Location Providera, poskytovat ostatním aplikacím, což bylo splněno.

V implementované aplikaci je určitě spousta věcí, které by se mohly doladit a vylepšit v dalším vývoji, jako je například optimalizace kódu pro ostatní verze operačního systému Android a rozšíření jazykové podpory o téměř v celém světě používanou angličtinu.

Použitá literatura

[1] BURNETTE, Ed. *Hello, Android*: *introducing Google's mobile development platform*. Version 2008-12-5. Raleigh, N.C.: Pragmatic Bookshelf, c2008, 218 s. ISBN 978-193-4356-173.

[2] MEIER, Reto. *Professional Android 2 application development*. Indianapolis: Wiley, c2010, 543 s. Wrox programmer to programmer. ISBN 978-0-470-56552-0.

[3] GPSd request/response protocol. *GPSd* [online]. [cit. 2012-04-30]. Dostupné z: http://www.catb.org/gpsd/gpsd_json.html

[4] EL-RABBANY, Ahmed. *Introduction to GPS*: *the Global Positioning System*. Boston, MA: Artech House, c2002, 176 s. ISBN 15-805-3183-0.

[5] *Android Developers* [online]. [cit. 2012-04-25]. Dostupné z: http://developer.android.com/

[6] NMEA data. *Navigation and GPS Articles* [online]. [cit. 2012-04-30]. Dostupné z: http://www.gpsinformation.org/dale/nmea.htm

[7] MURPHY, Mark L. *Android 2*: *průvodce programováním mobilních aplikací*. Vyd. 1. Brno: Computer Press, 2011, 375 s. ISBN 978-80-251-3194-7.

[8] *Symarctic Solutions Ltd.* [online]. [cit. 2012-04-30]. Dostupné z: http://www.symarctic.com/

[9] *Real time GPS Tracking solutions* [online]. [cit. 2012-04-30]. Dostupné z: http://gpsgate.com/

[10] *Bluetooth GPS* [online]. [cit. 2012-04-30]. Dostupné z: https://sites.google.com/site/googooandroid/btgps

[11] *GPSd4SE* [online]. [cit. 2012-04-30]. Dostupné z: http://blog.4zal.net/2009/06/22/gpsd4se-v0-1/

[12] About the NMEA. *NMEA* [online]. [cit. 2012-04-30]. Dostupné z: http://www.nmea.org/content/about_the_nmea/about_the_nmea.asp

[13] GPSD-NG: A Case Study in Application Protocol Evolution. *GPSd* [online]. [cit. 2012-04-30]. Dostupné z: http://gpsd.berlios.de/protocol-evolution.html

Seznam příloh

I. Implementovaná aplikace – Aplikace.zip
II. Uživatelská příručka – Uživatelská příručka.pdf
III. Programátorská dokumentace – Dokumentace.zip
IV. Testovací data – Data.zip